Automation of the Quantum Algorithm HHL for implementing two-dimensional SVMs

Gabriela Pinheiro[1], **Luis Antonio Brasil Kowada**[1]

[1]Instituto de Computação – Universidade Federal Fluminense (UFF) – Niterói, RJ – Brazil

gabrielapc@id.uff.br, luis@ic.uff.br

Abstract. *Support Vector Machine (SVM) is considered one of the main classification Machine Learning algorithms. Following the original formulation,an SVM generation has quadratic complexity, leaving room for exploring resolution methods with better performance. One way to enhance its efficiency is by utilizing Quantum Computing algorithms, such as the HHL. This work presents an automation of a Quantum Machine Learning algorithm that uses HHL to generate SVMs fixed at the origin of a two-dimensional hyperplane.*

1. Introduction

The Support Vector Machine(SVM) is considered one of the most powerful classification algorithms which, due to its strong theoretical foundations and generalization capability, is widely used in relevant applications such as bioinformatics and image classification [Cervantes et al. 2020]. The original formulation of the algorithm has a quadratic complexity. To reduce the complexity of the algorithm, [Suykens and Vandewalle 1999] applied a least-squares reformulation on the original version of the SVM, transforming it on the system of linear equations. This transformation allows for more efficient linear systems resolution techniques to be applied, such as quantum algorithms.

Quantum Computation allows the use of quantum mechanics to obtain advantages over classical computers on information processing on specific cases. Despite quantum computers with enough power to process large computations are not yet available, the field already demonstrates being promising, with algorithms having exponential performance advantages [Gyongyosi and Imre 2019].

HHL [Harrow et al. 2009] is a quantum algorithm where is possible to extract information about the solution of a linear system of equations with exponential advantage over classic algorithms. The algorithm is used in a variety of quantum machine learning applications, such as linear regression and SVMs [Duan et al. 2020].

The objective of this paper is to validate and automate an approach to the use of HHL in a Quantum Machine Learning algorithm proposed by [Rebentrost et al. 2014] to generate SVMs fixed at the origin of a two-dimensional hyperplane.

2. Fundamental Concepts

2.1. SVM

Support Vector Machine (SVM) [Cortes and Vapnik 1995] is a Machine Learning algorithm created originally for binary classification of data, were a dataset is divided in two distinct classes and the SVM needs to discover to which class a new element of the dataset belongs to.

The algorithm works by non-linear mapping the data to vectors in a hyperspace where the dimension is defined by the number of parameters used. The goal is to find a linear decision hyperplane dividing the data from both classes, maximizing the distance between them. As a result, the class of new data can be verified by it's position relative to the hyperplane.

The hyperplane to be found is represented by the equation $\vec{w} \cdot \vec{x} - b = 0$, and the classes are represented by the labels 1 and -1. A data $\vec{\chi}$ belongs to the 1 class if $\vec{w} \cdot \vec{\chi} - b \geq 1$ is true, or belongs to the -1 class if $\vec{w} \cdot \vec{\chi} - b \geq -1$ is true. Figure 1 shows an example of a SVM in a two-dimensional hyperspace.

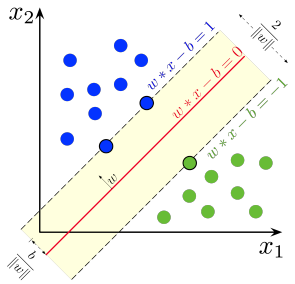

Figure 1. Maximum-margin hyperplane in a two-dimensional hyperspace. Source: [Larhmam 2018].

SVM's original formulation main goal is to maximize the distance margin $\frac{2}{||w||}$ between the two classes, which is equivalent to minimize $\frac{\|\vec{w}\|^2}{2}$ with the restriction $y_j(\vec{w} \cdot \vec{x_j} + b) \geq 1$ for every x_j in the dataset.

[Rebentrost et al. 2014] proposed a least-squares reformulation of the problem, transforming the SVM into a solution of the linear system represented by Equation (1).

$$F \begin{pmatrix} b \\ \vec{a} \end{pmatrix} \equiv \begin{pmatrix} 0 & \vec{1}^T \\ \vec{1} & K + \gamma^{-1} I \end{pmatrix} \begin{pmatrix} b \\ \vec{a} \end{pmatrix} = \begin{pmatrix} 0 \\ \vec{y} \end{pmatrix}, \tag{1}$$

where K the kernel matrix, a $M \times M$ matrix created using the function $K_{ij} = k(\vec{x_i}, \vec{x_j}) = \vec{x_i} \cdot \vec{x_j}$ with the M training vectors, being $\vec{y} = (y_0, y_1, ..., y_{M-1})$ and $\vec{1} = (1, ..., 1)$. I represents the identity matrix and γ is the user-defined training error weight. Resulting in a $(M + 1) \times (M + 1)$ matrix F in which the SVM parameters become represented as a function of b and $\vec{a}$ and the class of a data $\vec{x_i}$ is now obtained with the condition in (2).

$$y_i = \begin{cases} +1 \text{ if } \sum_{j=0}^{M-1} a_i k(\vec{x_i}, \vec{x_j}) + b \geq 0 \\ -1 \text{ if } \sum_{j=0}^{M-1} a_i k(\vec{x_i}, \vec{x_j}) + b < 0 \end{cases} \tag{2}$$

2.2. Quantum Computation

Quantum Computation is the field of computation responsible for studying information processing through the use of quantum mechanical systems [Nielsen and Chuang 2010], which makes possible to obtain algorithms with processing advantages over classical systems.

Named after the type of quantum computers available, Quantum Computation is currently at the NISQ (Noisy Intermediate-Scale Quantum) era, where these computers processing capacity is still limited, between 50 to 100 qubits [Pednault et al. 2017]. Another limiting factor is the high rate of noise, which are measuring errors that occurs due to different reasons, such as the the difficulty to maintain the system isolated, multi-qubit operations and the circuit transpilation that is necessary to run circuits on real devices and usually results in bigger circuits than the original.

2.3. HHL algorithm

Named after its creators, the HHL [Harrow et al. 2009] is a quantum algorithm for the resolution of linear systems. Considering the system $A\vec{x} = \vec{b}$, given $\vec{b}$ and the matrix A, it is possible to extract information about the solution $\vec{x}$ more efficiently, being exponentially faster than any classical algorithm for specific systems. A more in depth explanation and circuit tutorial can be found in [Adedoyin et al. 2018].

3. Implementation

In order to reduce the size and length of the generated circuit for a SVM, [Yang et al. 2019] fixed $b = 0$, implying that the generated hyperplane passes trough the origin of the hyperspace. This was done in order to simplify the system, because with b being 0 is possible to remove the first line and column of F due to their values also becoming 0 after the multiplication, resulting in the Equation (3).

$$F\left(\vec{a}\right) \equiv \left(K + \gamma^{-1} I\right)\left(\vec{a}\right) = \left(\vec{y}\right) \tag{3}$$

As a result, F becomes a $M \times M$ matrix and the only parameter that must be found is $\vec{a}$. Because $\vec{a}$ is a vector, the ratio between it's components is enough to calculate the angle and trace the hyperplane. Resulting in an ideal scenario for the use of HHL, which is able to calculate that ratio efficiently.

Quantum Machine Learning is an emerging area, therefore, the majority of the circuit generation needs to be done manually due to the lack of automated tools available, resulting in a previous quantum computation knowledge requirement from the user. The circuit generation automation not only increases the efficiency of the testing and development process but also removes the knowledge requirement.

Automating was performed using the Python programming language and the qiskit library, an open-source quantum toolkit developed by IBM [Cross 2018]. Each HHL circuit used the same structure shown in the tutorial [Morrell Jr et al. 2021], where initially $\vec{b}$ is mapped to the circuit, followed by a Quantum Phase Estimation(QPE), a controlled rotation R and finishes with a inverse QPE. The generated circuit is represented on Figure 2.

In order to validate the circuit automation, the Breast Cancer Wisconsin (Diagnostic) dataset [Street et al. 1993] was chosen to perform the tests. The dataset was created from 569 images of breast mass fine needle aspirates (FNA) that were processed and resulted in 30 continuous features.

After plotting the graphs for all possible combination of features pairs, the ones who presented the best behavior for an SVM implementation were empirically chosen to

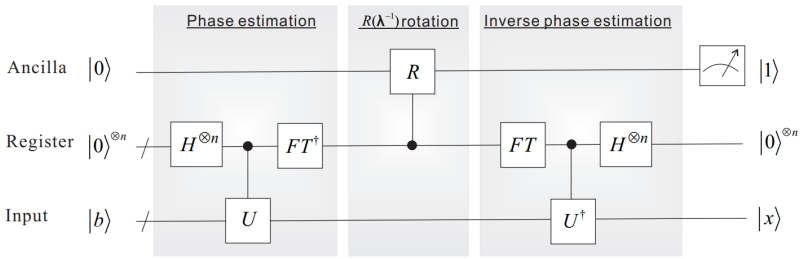

Figure 2. Generic HHL circuit. Source: [Cai et al. 2013].

compose the test group, which, due to the free plan execution time limit, was restrained to eleven different SVMs. For each test, the dataset was divided in a training set containing the first 369 elements and a test set with the remaining 200. The Kernel matrix was created with the average value of the features of the benign and malignant training data, and each element was divided by its norm.

4. Results

The generated SVMs were classically validated and the corresponding circuits were tested in quantum simulators and real quantum computers with the number of qubits varying from 7 to 127, chosen based on their availability, provided for free by IBM, with 100.000 measurements in every execution. For each element of the test group, three different SVMs were generated, one from the exact solution of the linear system, one from the results of noiseless simulations and one from executions on real quantum devices, evaluated based on their accuracy, calculated by $\frac{correct\ predictions}{all\ predictions}$.

In all test cases, the SVMs generated from the exact solution and simulations presented similar accuracy, with slightly variation due to numerical rounding performed by the algorithm, but the SVMs generated from executions on real quantum computers showed a expressive results variation due to the noise interference, due to time constraints it was not possible to carry out an in-depth analysis of the different origins of the noise and its impacts. Resulting in three distinct scenarios, positive noise interference, low noise interference and negative noise interference, each represented by a example above.

Occurring only once, in this scenario the noise interference improved the accuracy of the generated SVM, reaching 87.0% in comparison with the 83.0% expected. This experiment was executed on the retired 7 qubits *ibm_nairobi* computer and the results are shown in Figure 3.

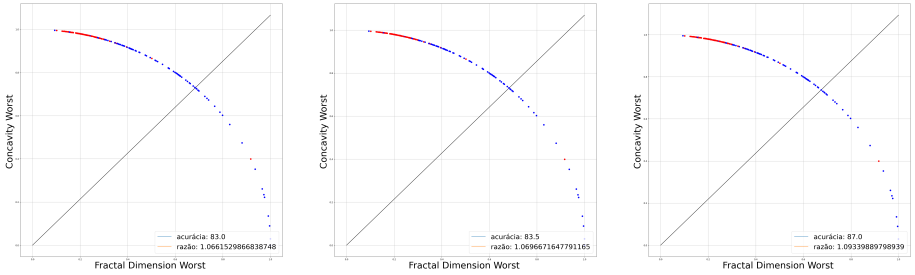

Figure 3. SVMs positively affected by noise, generated by the exact solution, simulation and execution on a real quantum computer, respectively.

Another rare case, in this scenario the noise interference only caused a small drop in accuracy, being 76.0% compared to expected 79.5%.This experiment was executed on the 127 qubits *ibm_brisbane* computer and the results are shown in Figure 4.

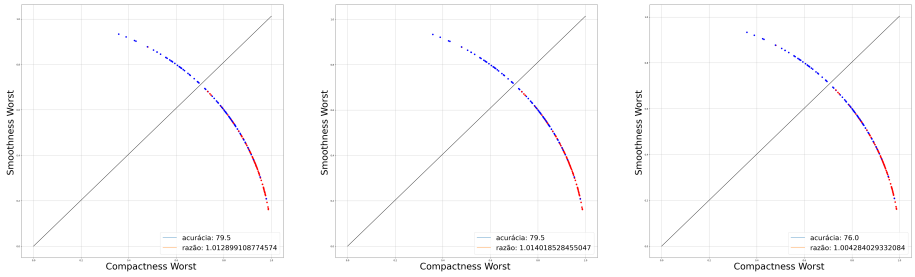

Figure 4. SVMs little affected by noise, generated by the exact solution, simulation and execution on a real quantum computer, respectively.

The most common case, in this scenario, the noise interference caused a major drop in the accuracy of the generated SVM, reaching only 67.5% in comparison with the 88.5% expected in this example. This experiment was also executed on *ibm_nairobi* and the results are shown in Figure 5.

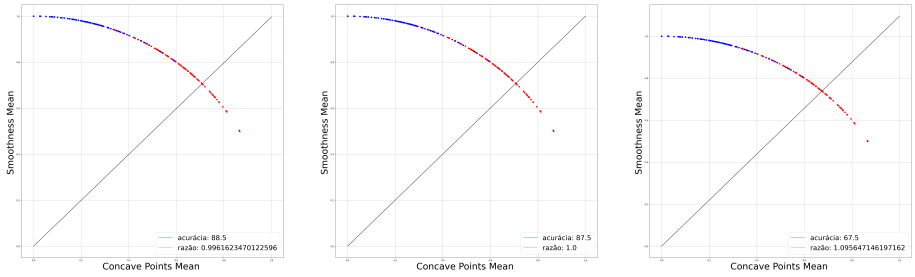

Figure 5. SVMs negatively affected by noise, generated by the exact solution, simulation and execution on a real quantum computer, respectively.

An important fact to take in consideration about the results on real quantum computers is that, due to the free plan's limit of execution time, each test case was only executed once, when the ideal scenario was to execute multiple times in order to explore the noise variation for each execution and search for better results.

5. Conclusion

The experiments showed results consistent with the expected solutions, demonstrating that the automation was done correctly. The circuit automation allows for a SVM to be generated from any two-dimension Kernel matrix automatically, facilitating the implementation of this Quantum Machine Learning algorithm for distinct applications. The accuracy obtained by the SVMs when executed on real quantum devices shows this method's viability, where the use of HHL for the linear system resolution could generate an efficiency gain over classical computations, indicating a possible quantum advantage.

In conclusion, this paper presented an automation algorithm for quantum machine learning to generate SVMs fixed on the origin of a two-dimensional hyperplane. From the two-dimensional SVMs automation, the next goal is the generalization for N-dimensional hyperplanes. Is expected that with bigger dimensions will be possible to obtain better results, because it will allow the use of more features simultaneously.

6. Acknowledgements

This work has been supported by the CNPq (101170/2023-8) and by the FAPERJ (260003/015313/2021).

References

A. Adedoyin et al. (2018). Quantum algorithm implementations for beginners. *arXiv preprint arXiv:1804.03719*.

X.-D. Cai et al. (2013). Experimental quantum computing to solve systems of linear equations. *Physical Review Letters*, 110(23):230501.

J. Cervantes et al. (2020). A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, 408:189–215.

C. Cortes and V. Vapnik (1995). Support-vector networks. *Machine learning*, 20:273–297.

A. Cross (2018). The ibm q experience and qiskit open-source quantum computing software. In *APS March meeting abstracts*, volume 2018, pages L58–003.

B. Duan et al. (2020). A survey on HHL algorithm: From theory to application in quantum machine learning. *Physics Letters A*, 384(24):126595.

L. Gyongyosi and S. Imre (2019). A survey on quantum computing technology. *Computer Science Review*, 31:51–71.

A. W. Harrow, A. Hassidim, and S. Lloyd (2009). Quantum algorithm for linear systems of equations. *Physical Review Letters*, 103(15):150502.

C. B.-S. . v. W. C. Larhmam (2018). Maximum-margin hyperplane and margin for an SVM trained on two classes. samples on margins are called support vectors. https://commons.wikimedia.org/wiki/File:SVM_margin.png.

H. J. Morrell Jr, A. Zaman, and H. Y. Wong (2021). Step-by-step hhl algorithm walkthrough to enhance the understanding of critical quantum computing concepts. *arXiv preprint arXiv:2108.09004*.

M. A. Nielsen and I. L. Chuang (2010). *Quantum computation and quantum information*. Cambridge University Press.

E. Pednault et al. (2017). Pareto-efficient quantum circuit simulation using tensor contraction deferral. *arXiv preprint arXiv:1710.05867*.

P. Rebentrost, M. Mohseni, and S. Lloyd (2014). Quantum support vector machine for big data classification. *Physical Review Letters*, 113(13):130503.

W. N. Street, W. H. Wolberg, and O. L. Mangasarian (1993). Nuclear feature extraction for breast tumor diagnosis. In *Biomedical image processing and biomedical visualization*, volume 1905, pages 861–870. SPIE.

J. A. Suykens and J. Vandewalle (1999). Least squares support vector machine classifiers. *Neural processing letters*, 9:293–300.

J. Yang, A. J. Awan, and G. Vall-Llosera (2019). Support vector machines on noisy intermediate scale quantum computers. *arXiv preprint arXiv:1909.11988*.